Paying Due Diligence to Software Architecture in Acquisition

Mike Gagliardi
Tim Morrow

Software Solutions Conference 2015
November 16–18, 2015
Copyright 2015 Carnegie Mellon University

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN “AS-IS” BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

This material has been approved for public release and unlimited distribution except as restricted below.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

DM-0002995
The average project spends 50-80% of its total effort correcting defects. How is that possible?
Most projects wait until the red zone to detect and correct their defects. Better-run projects find most defects in the blue zone.

Activity in which a Defect is Introduced

Quality Attributes

Drive
Architecture

Drives
Construction

Requirements Architecture Construction System Test Post-Release

Activity in which Defect is Detected

— Relative Cost to Correct Defects —

Average Cost to Correct a Defect

Source: Adapted from Construx; Software Development Best Practices; www.construx.com copyright © 2008 Construx
Why Is Architecture So Important?

Architecture is a common high-level communication vehicle for system stakeholders that is amenable to analysis and synthesis.

Architecture embodies the earliest set of design decisions about a system. These decisions

• are the most profound
• are the hardest to get right
• are most difficult to change
• ripple through the entire software development effort
• are most costly to fix downstream
• are critical to achieving mission/business goals

The earlier we reason about architecture tradeoffs, the better.
Why Is Architecture So Important?

The right architecture paves the way for system success.

First design artifact that addresses:
- performance
- reliability
- modifiability
- security

Key to systematic reuse:
- transferable, reusable, analyzable abstraction

Provides early low-cost means to predict system qualities:
- amenable to analysis and synthesis
- amenable to evaluation

The wrong architecture usually spells some form of disaster.
Characteristics of an Architecture-Smart Acquisition

- Understanding the mission drivers for the system being acquired
- Understanding quality attribute expectations of stakeholders
- Developing or selecting the software architecture
- Documenting and communicating the software architecture
- Analyzing and evaluating the software architecture
- Implementing the system based on the software architecture
- Ensuring that the implementation conforms to the software architecture
- Appropriately evolving the architecture over the system’s life cycle
- Incorporating other architecture-related management and development activities to achieve specific program objectives
Representation of Contract Performance Phase

- **Pre-Contract Work**
  - Government performs
  - Acquisition Planning and RFP/Contract Preparation
  - Requirements, CONOP, Reference Architecture, etc.

- **Contract Performance Phase**
  - Management Oversight and Technical Monitoring
  - Contractor Responsibilities
  - Technical Planning, Configuration Management, and Risk Management
  - Requirements Elaboration, Architectural Design, Detailed Design, Implementation, Test and Integration
  - -- Representative System and Software Development Activities --
  - Government performs
  - Post-Delivery Work

- **Iteration**
  - On-Going Interaction

- **Iteration**

---

Paying Due Diligence to Software Architecture
November 2015
© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Software Engineering Institute | Carnegie Mellon University
Representation of Contract Performance Phase

- Government performs Pre-Contract Work
- Management Oversight and Technical Monitoring
- Government performs Post-Delivery Work

Acquisition Planning and RFP/Contract Preparation

- Requirements, CONOP, Reference Architecture, etc.
- Be proactive and specify architecture approach up-front in RFP/contract

Contract Performance Phase

- Specifying tasks and required actions as being event driven
- Avoiding dependencies on contractor's specific development approach
- Ensure a coherent approach by incorporating key clauses in DIDs governing traditional acquisition documents

- Representative System and Software Development Activities
- Technical Planning, Architecture Design, Detailed Design, Implementation, Test and Integration

- Representative System and Software Development Activities

- Government performs On-Going Interaction

- Test and Acceptance and Operational Deployment
## Typical Scenario Describing Impact of Adopting an Architecture-Smart Acquisition Approach

<table>
<thead>
<tr>
<th>BEFORE</th>
<th>AFTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>There is no software architecture documentation.</td>
<td>A documented software architecture is a contract deliverable.</td>
</tr>
<tr>
<td>The system’s non-functional (i.e., quality) requirements that greatly impact the architecture design and software implementation are poorly defined.</td>
<td>The system’s quality requirements are specified in terms of a clear and concise set of quality attribute scenarios generated by key stakeholders.</td>
</tr>
<tr>
<td>The development contractor presents a couple of PowerPoint box-and-line drawings to describe the architecture and high-level software design.</td>
<td>The software architecture description includes a comprehensive set of views (e.g., module decomposition, allocation, run-time) that is amenable to analysis.</td>
</tr>
<tr>
<td>The proposed software design is not appropriately analyzed or evaluated.</td>
<td>The software architecture is evaluated with stakeholder participation and risks (and risk themes) are identified and appropriately documented.</td>
</tr>
<tr>
<td>Architecture development is ad hoc and not based on careful analysis.</td>
<td>As a result of the architecture evaluation, the development contractor creates a risk mitigation plan and presents it at the Preliminary Design Review (PDR).</td>
</tr>
<tr>
<td>Plans for architecture evolution are ad hoc and not based on careful analysis.</td>
<td>In conjunction with the risk mitigation plan the development contractor develops a software architecture improvement roadmap based on an incremental software development approach.</td>
</tr>
</tbody>
</table>
Contractual View of DoD Acquisition Life Cycle

Each contract has a different objective and scope of work, but common elements

**PHASES**
- Materiel Solution Analysis
- Technology Development
- Engineering and Manufacturing Development
- Production and Deployment
- Operations and Support

**CONTRACTING**
- Study Contracts
- Technology Development Contract
- System and Software Development Contract
- Low-Rate Initial Production Contract
- Production Contract
- Post-Production Software Support Contracts
- Sustainment Contracts

**Milestones**
- A
- B
- C

**Pre-Systems Acquisition**
- Acquisition Planning
- RFP / SOW
- Source Selection

**Systems Acquisition**
- Contract Performance Phase with Government Oversight

**Sustainment**
- System Delivery and Acceptance
Contractual View of DoD Acquisition Life Cycle

– *Adopting an Architecture-Smart Acquisition Approach* –

**PHASES**

- Materiel Solution Analysis
- Technology Development
- Engineering and Manufacturing Development
- Production and Deployment
- Operations and Support

**Milestones**

- A
- B
- C

**CONTRACTING**

- Pre-Systems Acquisition
- Systems Acquisition
- Sustainment

- Study Contract
- Technology Development Contract
- System Development and Demonstration Contract
- Low-Rate Initial Production Contract
- Production Contract
- Post-Production Software Support Contracts
- Sustainment Contracts

**Requires taking appropriate action in these phases**

**Proactive**

**Reactive**

© 2015 Carnegie Mellon University

Distribution Statement A: Approved for Public Release; Distribution is Unlimited
Two Fundamental Ways Architecture-Smart Activities can be Incorporated in an Acquisition

Reactive

Software Architecture activities are *initiated opportunistically* and performed in situ under an existing contract at the request of the program manager.¹

Proactive

Software Architecture activities are *preplanned and integrated* up front in a request for proposal (RFP) for a system (or software) acquisition.

¹ Or at the request of a contractor under a negotiated agreement.
Integration of Systems and Software Engineering Aspects in an RFP

the way it’s commonly done today

the “integrating element” of the system & software engineering aspects in the traditional RFP

RFP Preparation

Systems Engineering Section of RFP
Promoting System and Software Engineering Congruency in Acquisition

Definition of Synergy
1. The interaction of two or more agents or forces so that their combined effect is greater than the sum of their individual effects.
2. Cooperative interaction among groups, especially among the acquired subsidiaries or merged parts of a corporation, that creates an enhanced combined effect.
Promoting System and Software Engineering Congruency in Acquisition

FROM the “staple” paradigm

the “integrating element” of the system & software engineering aspects in the traditional RFP

TO a synergy-driven paradigm

RFP Preparation

How can you help achieve more synergy and cooperation between systems and software engineering?

• What can you do on the acquisition organization’s side-of-the-fence?
• What can you do during on the development contractor’s side-of-the fence—i.e., during the contract performance phase?
Promoting System and Software Engineering Congruency in Acquisition

FROM the “staple” paradigm

the “integrating element” of the system & software engineering aspects in the traditional RFP

TO a synergy-driven paradigm

An Overarching

Use Cases for functional requirements

Quality Attribute Scenarios for non-functional requirements

System Context Diagram

System & Software Architecture Evaluation

Conduct

Adopt

Create

Quality Attribute Workshop

Software Engineering

System Engineering

Conduct

RFP Preparation
Key Elements of an Architecture-Smart Acquisition and Development Approach

1. Specifying a system’s quality attributes

   This involves conducting a Quality Attribute Workshop (QAW) with key stakeholders to elicit and capture\(^1\) quality attribute scenarios (i.e., specify the non-functional requirements) so the architecture can be appropriately designed.

2. Evaluating the system and software architecture

   This involves conducting an architecture evaluation (in collaboration with the system developer) using the system variant of the SEI’s Architecture Tradeoff and Analysis Method to identify and mitigate risks early in the system development cycle.

\(^1\) To represent and record in a lasting form
Conceptual Flow

Such a “big picture” view of a contractor’s architecture development approach would be described in its Software Development Plan (SDP).
## Conceptual Flow of ATAM

1. **Business Drivers**
2. **(System & Software Architecture)**
3. **Quality Attributes**
4. **Architectural Approaches**
5. **Scenarios**
6. **Architectural Decisions**

**Impacts**

- **Tradeoffs**
- **Sensitivity Points**
- **Non-Risks**
- **Risks**

**Distilled into**

**Risk Themes**

**Qualitative Analysis**
Elements of an Architecture-Smart Acquisition

Legend
- APW – Acquisition Planning Workshop
- PDR – Preliminary Design Review
- QAW – Quality Attribute Workshop
- SRR – System Requirements Review
- SWARD – Software Architecture Description Document
- CDR – Critical Design Review
- ATAM – Architecture Trade-off Analysis Method

N days after contract award
W days before ATAM
X days before PDR
Z days before CDR

What should happen Post-CDR???
Model for Incorporating Software Architecture Evaluation in an RFP/Contract

Short paragraph in SOW specifying a software architecture evaluation is to be conducted in accordance with a prescribed evaluation plan.

The plan is tailored to satisfy the program’s needs and placed in the government RFP/Contract Reference Library.

* See http://www.sei.cmu.edu/library/abstracts/reports/09tn004.cfm for an example description of an ATAM plan.

Instantiation → Elements of an Architecture-Smart Acquisition Approach
## Ensuring a Coherent Approach is Adopted

<table>
<thead>
<tr>
<th>Document</th>
<th>Type of Information to Be Included (Relative to Conducting an Architecture Evaluation)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEMP</td>
<td>Describe: (1) how the architecture evaluation is integrated into the system engineering management plan in relation to the program milestones, (2) how the system’s quality attribute requirements (i.e., nonfunctional requirements) that drive the architectural design will be specified and managed, and (3) how the software architecture will be documented.</td>
</tr>
<tr>
<td>TEMP</td>
<td>Describe the role of architecture evaluation in the test and evaluation management plan and when the evaluation will be scheduled in relation to the program milestones.</td>
</tr>
<tr>
<td>SEP</td>
<td>Describe: (1) how the architecture evaluation is integrated into the system engineering plan in relation to the system engineering milestones, (2) how the system’s quality attribute requirements (i.e., nonfunctional requirements) that drive the architectural design will be specified and managed, and (3) how the software architecture will be documented.</td>
</tr>
<tr>
<td>SDP</td>
<td>Describe how the software architecture evaluation fits into the overall software development approach including how identified risks (and risk themes) will be analyzed and mitigated.</td>
</tr>
<tr>
<td>STP</td>
<td>Describe the role of architecture evaluation in the software test plan and when the evaluation will be scheduled in relation to software testing milestones.</td>
</tr>
<tr>
<td>RMP</td>
<td>Describe how risks (and risk themes) emanating from the architecture evaluation will be integrated with the program’s risk management system and subsequently managed (i.e., identified, tracked, and mitigated).</td>
</tr>
</tbody>
</table>
Example: Architecture Aspects You May Want the Offeror to Describe in their Technical Proposal

Section L – Instructions to Offerors

1. Describe how quality attribute scenarios resulting from the QAW will be integrated into the requirements baseline and managed from that point forward.

2. Describe how architecture risks and risk themes discovered during the ATAM evaluation will be prioritized and mitigated.

3. Describe how proposed software modifications (including architectural changes) that occur during the system life cycle will be managed.

4. Describe how compliance of the software implementation with the approved software architecture baseline will be enforced throughout the life cycle.

5. Describe what kind of software architecture metrics will be collected and reported to the government during the contract performance phase.
Examples of Architecture Practices that Can Be Incorporated in an RFP/Contract

Architecture activities, deliverables and measures that can easily be incorporated into an RFP today for a system acquisition include:

- **Quality Attribute Workshop (QAW) to collaboratively**
  - Validate business and mission drivers
  - Elicit quality attributes scenarios (system and software)
  - Refine a set of quality attribute scenarios

- **Architecture design and evolution guidance**
  - Quality attribute-driven architectural design

- **Software architecture description**
  - Include as part of contractual deliverables

- **(System &) Software Architecture Evaluation**
  - Specify collaborative evaluation based on ATAM
  - Require evaluation report identifying risks and risk themes

- **Risk mitigation "monitoring instrument"**
  - Monitor the risk mitigation activities and report on progress

- **Cost benefit change analysis**
  - Prioritization of architecture risk mitigation activities based on cost benefit

So how do you decide what is right for Your program?
Conducting an Acquisition Planning Workshop

Why hold a workshop?

1. To be **proactive** and provide upfront assistance during the acquisition planning and RFP preparation phase when it can make a difference.

2. To provide a **structured forum** for key acquisition stakeholders to **understand** the program’s acquisition approach and current status, **and explore** potential ways for **reducing software acquisition risk** via a facilitated technical interchange

Outputs

1. Common understanding of the acquisition challenges, risks, and key issues

2. A list of actions for going forward with acquisition planning
Overview of Acquisition Planning Workshop

Understand → Elicit → Explore → Focus

- Program Overview
- Status of Acquisition Plans and Strategy
- Acquisition Timeline

Drivers and Constraints

- Risks and Issues
- Specific Acquisition Challenges
- Traditional Acquisition Approaches
- Alternative Acquisition Strategies

Impact of Lessons Learned

Risks, Issues and Acquisition Considerations

Action items and Next Steps

Architecture-smart acquisition practices for reducing risk

Specific Acquisition Challenges

Risks, Issues and Acquisition Considerations

Action items and Next Steps
How Acquisition Programs Can Leverage an Architecture-Smart Approach to Reduce Risk

Realize that Architecture is Key
• Embodies the early design decisions that are the most difficult to get right
• Provides level-of-abstraction best aligned with program responsibilities

Focus on Quality Attributes
• Allows stakeholders to discuss, clarify, and prioritize non-functional requirements that are often problematic

Acquire Architecture Documentation
• Provides the means to analyze the software design and guide development

Evaluate the System and Software Architecture
• Promotes coordination between system and software engineering

Focus on Risk Management
• Risk identification and mitigation

Arrange for Training
• Educate both program office and contract personnel

Conduct an Acquisition Planning Workshop
• Be proactive and endure the right stuff gets in the RFP/contract
Contact Information

Mike Gagliardi
Principal Engineer
Software Engineering Institute
Office: 412-268-7738
Email: mjg@sei.cmu.edu